LDAP-based Ontology for Information Integration

BTW 2001
Oldenburg, Germany

Lule Ahmedi[1], Pedro José Marrón and Georg Lausen
Institut für Informatik
Universität Freiburg
{ahmedi,pjmarron,lausen}@informatik.uni-freiburg.de

[1] work supported by the Deutsche Forschungsgemeinschaft, Aktenzeichen La 598/4-1.

Talk Outline

- Introduction
- LDAP Representation Formalism for Ontologies
- What about the integration?
- An Example Ontology
- Conclusion

Introduction

The Problem: Expanding heterogeneous data repositories in Internet

Up to date Solutions:

$KR\ meets\ DB \rightarrow$ **Ontologies**.

$\swarrow$ $\searrow$

an explicit specification of a conceptualization of the world of interest [Gruber 93]

a common query domain (interface) for all users of a heterogeneous repository

Our Approach:

$Directories\ meet\ DB \rightarrow$ **Directory-based Ontologies**.

Motivation:

- ▷ LDAP's popularity since conception of LDAP v3 protocol '97
- ▷ LDAP's technical grounds - its tight connection to network & distribution channels
- ▷ LDAP simple, coherent & uniform model.

... promises to be able to easily solve many classical integration problems along with the LDAP advantageous features.

LDAP Overview

LDAP stands for ”Light Weight Directory Access Protocol”.

LDAP server is a ss DB with limited transaction support, composed of:

Directory Schema: is a set of classes $Class(AttrName, AttrType)$ and determines:

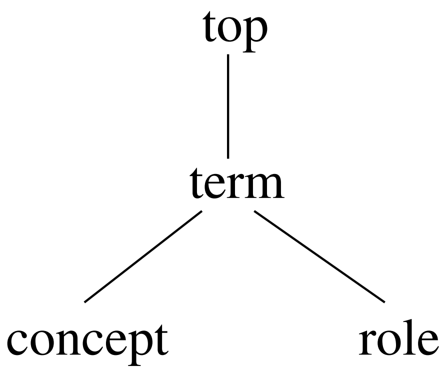
- ▷ how classes (their instances) can be structured on a tree - `class hierarchy`,
- ▷ the attributes that a given class instance must or may contain - `class content`.

Directory Instance: is a set of entries $Entry(AttrName, AttrValue)$

- ▷ belonging to at least one class (oc),
- ▷ placed in the instance hierarchy based on their dn, and
- ▷ having a structure conforming to the schema definition.

We use directory schema primitives (classes & attributes) to define ...

LDAP Representation Formalism for Ontologies



```
CLASS term
   MUST CONTAIN {OBJECT-CLASS oc, STRING name}
CLASS concept
   MAY CONTAIN {DN(concept) subclass_of,
                DN(concept) superclass_of,
                LDAP-Filter filter,
                DN(source term) link,
                DN(concept) synonym,
                ...}
CLASS role
   MUST CONTAIN {DN(concept) domain,
                 (DN(concept) or STRING) range}
   MAY CONTAIN {{YES} key,
                DN(source term) link,
                DN(role) synonym,
                ...}
```

What about the integration?

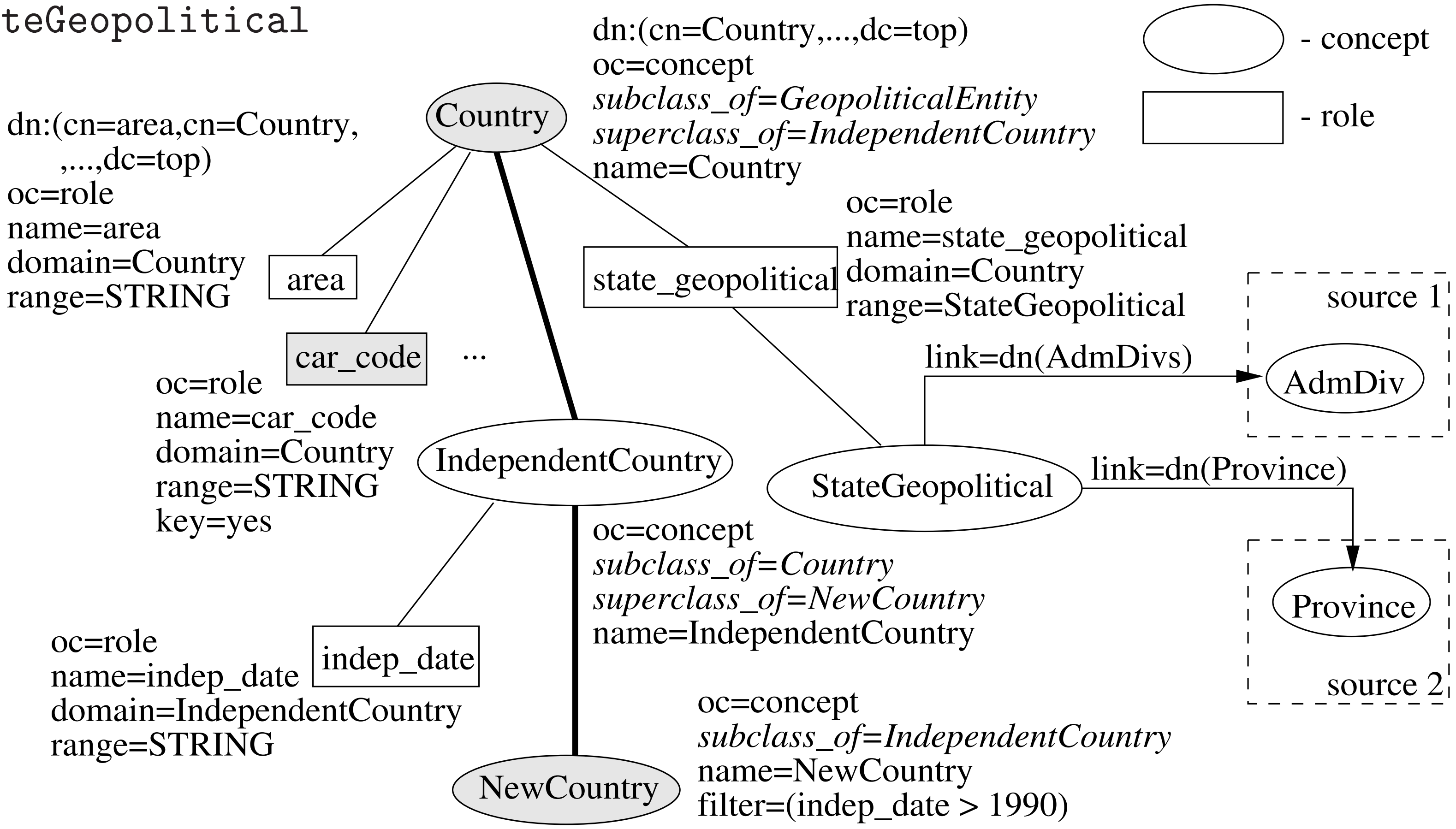
`link` attribute.

Specification: $t_o(i) : link = t_s(j).$

Semantic: indicates that a given `term` in ontology is semantically equivalent with the term in the LDAP source it references to.

An Example Ontology

```
Country(area,car_code,state_geopolitical)
  IndependentCountry(indep_date)
    NewCountry
StateGeopolitical
```



dn:(cn=Country,...,dc=top)
oc=concept
subclass_of=GeopoliticalEntity
superclass_of=IndependentCountry
name=Country

dn:(cn=area,cn=Country,
,...,dc=top)
oc=role
name=area
domain=Country
range=STRING

oc=role
name=state_geopolitical
domain=Country
range=StateGeopolitical

oc=role
name=car_code
domain=Country
range=STRING
key=yes

oc=concept
subclass_of=Country
superclass_of=NewCountry
name=IndependentCountry

oc=role
name=indep_date
domain=IndependentCountry
range=STRING

oc=concept
subclass_of=IndependentCountry
name=NewCountry
filter=(indep_date > 1990)

Conclusion

▽ Enables a *seamless* integration of source data, schemata discrepancies, and semantic information under a common framework.

▽ Is the only system (to our knowledge) that combines the advantages of:

▷ a hierarchical data model, suited in particular for mapping XML documents [YAT, MIX, TSIMMIS, FLORID],

▷ domain ontologies [ONTOBROKER, MOMIS, OBSERVER, SIMS] and

▷ network features of directory services.

▽ Development on it continued for making the following additions:

▷ query rewriting operators,

▷ schematic reconciliation and

▷ ontology partitioning.